Los Alamos
NATIONAL LABORATORY
EST. 1943

LA-UR-21-23834
Approved for public release; distribution is unlimited.

Title: Sesame ASCII 2 File Format

Author(s): Young, Ginger Ann

Intended for: Documentation of LANL software package to inform developers the details of a new file format.

Issued: 2021-04-20

Disclaimer:
Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by Triad National Security, LLC for the National Nuclear Security Administration of U.S. Department of Energy under contract 89233218CNA000001. By approving this article, the publisher recognizes that the U.S. Government retains nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

Sesame ASCII 2 File Format

Ginger Young
April 13, 2021

Table of Contents

The examples below illustrate a table or ASCII 2 formats and are not a complete material. The materials and tables in the examples are all fictitious.

1.0 Sesame ASCII 2 File Format

The target audience for this document is for SES_IO developers.

The first documented version of Sesame ASCII File format is a human readable file that was developed on 80-character punch cards. It is a fixed-format file with no spaces between the data and each data having a fixed number of characters. This document is 'SESAME: The Los Alamos National Laboratory Equation of State Database', LANL Report LA-UR-92-3407 (1992).

To improve the readability and ease of parsing a Sesame ASCII file a new version of the format was developed and it is described in this document. All versions of Sesame ASCII files are UNIX file types, where only a linefeed is supported as an end of a line demarcation.

There are a number of values that are no longer needed or supported for this version. Removed from this format are the following fields:

- r (an unused flag)
- 10000 to 11111 checksum values at the end of each numerical data line.

To increase flexibility, readability and parsing of the file, the following has been modified:

- A version number must be on the first line of each file: Version 2.0. The value of the version indicates the ASCII file format version and not the version of the contents of the file.
- Spaces or tabs must be between data.
- File_Number is only at the beginning of the definition of a table instead of at the beginning and end, and removing the need for an 'end of file' File_Number (which has the value of 2). Remember: File_Number has the values:
  - 0 – beginning of a material's first table description, and
  - 1 – for the next table description for the same material.
- Material_ID, Table_ID, and Number_of_Words are no longer 6 characters long; all can be the size of C programming language's long integer. The maximum value of a long int can be attained by writing a small program to print LONG_MAX while including: #include <limits.h>. At the time of writing this document, the maximum of a long int is 9223372936854775807.
- Comment Table's data definition remains the same. The data must be 80 ASCII standard characters/line until the last line of the comment, which may be less than 80 characters.
- Create_Date and Update_Dates are following ISO 8601 Date Standard format: YYYYMMDD.
- Version is no longer 4 characters in length; it is the size of C programming language's long integer .
- A line containing material data must not be longer than 160 characters.

1.1 Definition of a Material

The Sesame data library contains both thermodynamic (e.g., equation of state) and transport coefficients (e.g., opacity and conductivity). Note, for simplicity, the term EOS (equation of state) used herein includes both thermodynamic variables and transport coefficients. A Sesame ASCII 2 file is structured format to contain data corresponding to EOS for one or more materials. Each material contains one or more tables, and these tables contain the data. A table must have a line specifying the definition of its data and then one to many lines of data. The first table for each material must be a 101 table, which contains general information about the material. The requirements for 101 tables are described below in Requirements for 101 Tables.

A Sesame ASCII 2 file may have more than one material in it. A material must be completely defined before the next material's information is given. Materials and their tables cannot be interspersed. Each material must have a unique material identity, and all of the material's tables will use the same material identity.

For each material, the tables must be in ascending order and grouped together. Although this requirement is new, placing tables in ascending order was a standard practice. Some tools using Sesame data require the data to have tables in ascending order. The exception for this rule is the 100 table, which is used internally in SES_IO, and only created using SES_IO's user interface.

2.0 Definition of a Table

Each table has a record description and line(s) of data. The record description contains information to identify the material and table the data, when the data was created and last updated, a version number, and the Number_of_Words in the data. The record description must be the first line of a table and is composed of six values on a single line. These values must exist in the same order as described in the table below, and each value must have at least one space between them. One or more spaces are used as a demarcation between the values. The first value starts at the beginning of a line going left to right.

Order	Value Names	Description	C Language Type
1	File_Number	Beginning or continuation of a material	Long integer
2	Material_ID	Material Identification	Long integer
3	Table_ID	Table Identification	Long integer
4	Number_of_Words	Total number of data items. In 10*x* tables, this number is the number of characters in the definition. All other tables, the value is the number of numerical data.	Long integer
5	Create_Date	Date the table is created, where the format is YYYYMMDD	Integer
6	Update_Date	Date the table is updated, where the format is YYYYMMDD	Integer
7	Version_Number	An integer version of the material.	Long integer

The description line appears like the following in a Sesame ASCII 2 file:

File_Number Material_ID Table_ID Number_of_Words Create_Date Update_Date Version_Number

The Material_ID is unique for the material and is used for all of the material's tables. A material's tables are in consecutive and ascending order, except a 100 table which is generally at the end of a material. Once all of a material's tables are written to a file, a new material may start. A file can only contain one copy of a material.

If you are familiar with the original Sesame ASCII format LA-UR-19-24891, you can see that there are multiple changes. One of these changes is that the format for the dates has been changed. The old dates (Create_Date and Update_Date) had the format: MMDDYY. A decision was made to follow to ISO 8601 Date Standard, and update the format to YYYYMMDD.

Another change is File_Number, which used to be at the beginning and end of a record description. This version only has a file identifier at the beginning of each record description. File_Number has the value of '0' (zero) on the description at the beginning of a material, each subsequent table for a material has the value of '1' (one). The end of file description of two '2' value File_Numbers is no longer required.

The value Number_of_Words has a different meaning depending upon which type of table it is in. There are two types of tables: one gives comment information about the material, which contains ASCII text, and the other contains numerical data.

Comment tables always have a Table_ID from 101 to 199 or from 10101 to 10199. A 101 table must be present for each material and has its own requirements.

The data in a comment table contains human readable information about the material, and the data is composed of standard ASCII characters. The data must not contain control characters, Unicode, or characters that are not in the hex range of 20 to 7E and 09 (tab). If a comment table contains 80 or less characters, the characters must be on one line (i.e. the comment table contains one line of characters). If the comment contains more than 80 characters, each 80-character chunk is on a line until the remaining characters which are on the last line of the comment table.

When calculating the Number_of_Words for a comment table keep in mind that you do not add the linefeeds at the end of each 80-character chunk. The Number_of_Words is equal to the total number of characters including white spaces (a space or tab). **Remember**: spaces/tabs at the end of the last visible character needs to be added to Number_of_Words.

Numerical data tables must have a Table_ID larger than 199 and not between 10101 to 10199. These tables contain the material's numerical data, and Table_IDs are described in **LA-UR-92-3407 The Los Alamos National Laboratory Equation of State Database.** This reference also outlines how the data is organized in each table. In **LA-UR-19-30186 Sesame I/O Library User Manual Appendix B** there is a list and descriptions of Table_IDs supported by SES_IO. It is beyond this document to define table contents except for User-Defined tables.

Note: Table_IDs also have unique values in a material. Each material must not have a repeated Table_ID. If you have used a Table_ID of 101 and you want another comment table, you must increase the Table_ID by one for a new comment table. Which means Table_IDs are in consecutive and increasing order: 101, 102, 103…199.

A material's numerical data is written in **scientific notation's E notation**: 1.2345E-01, -1.2345E+10, +1.2345E-09, and Sesame data supports three data lengths: long (28 characters), medium (22 characters), and short (15 characters). The number of characters for each data length is spilt:

Data Length	**Sign**	**Mantissa**	**E**	**Sign**	**Order of magnitude**
Long	1	23	1	1	2
Medium	1	17	1	1	2
Short	1	10	1	1	2
Valid Characters	'+', '-', ' '	n.xxxx…x where: n: 1-9 x: 0-9	E	'+', '-'	xx where x: 0-9

Examples of each:

Data Length	**Values:**
Long	1.34567890123456789012 3E-99
Long	+1.34567890123456789012 3E+00
Long	-1.34567890123456789012 3E+99
Medium	1.34567890123456E-99
Medium	+1.34567890123456E+00
Medium	-1.34567890123456E+99
Short	1.34567890E-99
Short	+1.34567890E+00
Short	-1.34567890E+99

The numerical data follows a numerical table's record description and is Number_of_Words long. The data must not be more than five numbers per line, and data must have five numbers per line until the last line of data for that table. The last line contains up to five numbers corresponding to the remainder of the table's data. For instance, if there are 25 numbers in a numerical table, there is 5 lines of data with 5 numbers on each line. If data has 32 numbers, the first 6 lines has 5 numbers, while the last line has 2 numbers.

Note: LANL's Sesame IO Library, SES_IO, can only read/write **medium** and **short** Sesame ASCII data.

3.0 Table Examples

3.1 Comment Table (Table_ID >= 101 and Table_ID <= 199) or (Table_ID >= 10101 and Table_ID <= 10199)

A comment table has the Table_ID = 101 to 199 or 10101 to 10199.

The following comment table is an example for a 102 table and contains fictional data. The first line of a table is the record description. This record description contains information for the File_Number, Material_ID, Table_ID, Number_of_Words, Create_Date, Update_Date, and Version in that order. Space(s) or tab(s) is used as a delimiter.

This example's 102 table description is

File_Number: 1, (this is not the first table for material 3720)
Material_ID: 3720,
Table_ID: 102,
Number_of_Words: 107,
Create_Date: 20200414 for April 14, 2020,
Update_Date: 20200414 for April 14, 2020, and
Version: 1.

The lines following the description contain the table's data. In the case of a comment table, the data comprises of Standard ASCII characters and no control characters beyond a linefeed or a tab. These lines may not be longer than 80 characters excluding the linefeed.

The following diagram outlines the lines of text in this table, and the number of characters in each of the lines of data. The total number of characters is how we calculate the Number_of_Words; do not include newlines.

Num Words	1st line:	1 3720 102 100 20200414 20200414 1
80	2nd line:	There needs to be 100 characters in this table. The number includes all of the
20	3rd line:	spaces in each line.

This is how the data looks in a Sesame ASCII 2 file:

```
1 3720 102 100 2020414 2020414 1
There needs to be 100 characters in this table. This number includes all of the
spaces in each line.
```

The next example is a 101 table. A 101 table is required to contain expected keywords. In this example, each keyword is on a separate line to illustrate the keywords. Multiple keywords on a line are allowed. For this example, our description contains:

File_Number: 0 (it is the first table for material 3720),
Material_ID: 3720,
Table_ID: 101,
Number_of_Words: 194,
Create_Date: 20170823 for August 23, 2017,
Update_Date: 20200421 for April 21, 2020, and
Version: 2.

The following diagram outlines the lines of text in this table, and the number of characters in each of the lines of data. This total number of characters is how we calculate the Number_of_Words.

Num Words	1st line:	0 3720 101 194 20170823 20200421 2
80	2nd line:	material aluminum (z=13.0, a=26.9815)/ source. Frodo Baggins, Bag End/ date. Aug
80	3rd line:	. 23, 2017/ refs. LAUR-ZX1/ comp. Al/ codes. Nazgul/ classification. Unclassifie
34	4th line:	d/ other. This data is fictional./

In a Sesame ASCII 2 file this data looks like:

```
0 3720 101 194 20170823 20200421 2
material aluminum (z=13.0, a=26.9815)/ source. Frodo Baggins, Bag End/ date. Aug
. 23, 2017/ refs. LAUR-ZX1/ comp. Al/ codes. Nazgul/ classification. Unclassifie
d/ other. This data is fictional./
```

3.2 Numerical Table Table_ID > 199 and not (Table_ID >= 10101 and Table_ID <= 10199))

A numerical table's Table_ID must be greater than 199 and not be between 10100 and 10200. These Table_IDs are either for comment tables or have restricted use.

The following example is for a numerical table, 201. In this table's record description is
File_Number is 0 (first description for material 2030),
Material_ID is 2030,
Table_ID is 201,
Number_of_Words is 5,
Create_Date is 19931221 for December 21, 1993,
Update_Date is 19940607 for June 7, 1994, and
Version is 1

The first line of the table's record description contains the above information in left to right order: File_Number, Material_ID, Table_ID, Number_of_Words, Create_Date, Update_Date, and Version. Each field must be delimited by one of more spaces.

Following the table's record description are lines of numerical data for the table. In this example there are five words of data, and since there can only be five words per line, there is only one more line in this table.

The figure below outlines the first line of the table, and then the five words of data in the second line.

Num Words	1st line, all words:	0 2030 201 5 19931221 19940607 1
1	2nd line, 1st word	1.999999999999989E+01
2	2nd line, 2nd word	4.007999999999970E+01
3	2nd line, 3rd word	1.546999999999976E+00
4	2nd line, 4th word	0.000000000000000E+00
5	2nd line, 5th word	0.000000000000000E+00

This is how the above looks in a file:

```
0 2030 201 5 19931221 19940607 1
1.999999999999989E+01 4.007999999999970E+01 1.546999999999976E+00 0.000000000000000E+00 0.000000000000000E+00
```

The next example shows a Number_of_Words not divisible by five and with the data length of short. This example is a 303 table, and its record description is

File_Number: 1 (this is not the first description for material 2030),
Material_ID is 2172,
Table_ID is 303,
Number_of_Words is 33,
Create_Date is 19920304 for March 4, 1992,
Update_Date is 19960714 for July 14, 1996, and
Version is 6.

In a Sesame ASCII 2 formatted file, the data appears:

```
1  2172     303     33   19920304    19960714   6
 3.00000000E+00  3.00000000E+00  6.40900000E+00  6.40900000E+00  6.40900000E+01
 6.23420000E+00  6.23420000E+00  6.23420000E+00  6.32000000E+01  6.32000000E+01
 6.32000000E+01  6.32000000E+01  6.32000000E+01  6.32000000E+01  6.32000000E+01
 6.32000000E+01  6.32000000E+01  6.34000000E+01  6.34000000E+01  6.34000000E+01
 6.34000000E+01  6.34000000E+01  6.34000000E+01  6.34000000E+01  6.34000000E+01
 6.34000000E+01  6.36000000E+01  6.36000000E+01  6.36000000E+01  6.36000000E+01
 6.36000000E+01  6.36000000E+01  6.36000000E+01
```

4.0 Requirements for 101 Tables

Each material is required to have a 101 table, which is a comment table. Historically this has not been enforced, and there are many existing material files that do not have comment tables.

A 101 table should contain the following keywords: material, source, date, refs, comp, codes, and classification. The keyword 'material.' must be in the comment and must be the first keyword used. If the material is multiphase, it must contain the keyword 'phases.'. The keyword 'other.' can be used for any other information.

All of the keywords are lowercase and must have a period ('.') at the end of each word. Following the keyword is the keyword's text followed by a '/'. A '/' must be at the end of each keyword's text.

Keyword	Current Convention	Example
material.	Material name (z=atomic number, a=atomic weight)	material. Aluminum (z-13.0, a=26.9815)/
source.	Author, organization	source. J. Edgar Hoover, Zx-10/
date.	Date material was created	date. Aug. 28, 1998/
refs.	Published references	refs. LAUR-99-9090/
comp.	Composition of the material	comp. Al /
codes.	Programming code used to create the material	codes. Foxtrot Express (ver. 1.40a)/
classification.	Classification of the material	classification. Export Controlled/
phases.	For multiphase material.	phases. Multiphase example/
other.	Any other information you wish to add	other. This is just an example, and has no real data./

5.0 Requirements for User-defined Tables

User-defined tables are only created through SES_IO's user interface and only written or read in binary, ASCII 1 & 2 file formats. The creation of a 100 table is done through a series of SES_IO library commands. The placement of the table is done internally in SES_IO, and one should not attempt to create one by hand. The 100 table mostly follows the definition in 2.0 Definition of a Table.

For ASCII 1 & 2 and binary file formats, the C-language sizes are restricted to the length for those formats. For instance, ASCII 1 file format's Material_ID and Table_IDs must be no longer than 6 characters; whereas ASCII 2 is limited by the size of C's long integer.

All three file formats' Create_Date and Update_Dates use the YYYYMMDD structure. The Number_of_Words is correctly calculated for whichever file format is being used, so the user need not worry. A table 100's version number is always 1.

The data in the 100 table is used in SES_IO internals and contains values for creating SES_IO constructs. The data contains ASCII character data. There may be multiple user-defined tables for each material, and SES_IO uses the '@' character as a delimiter between each. Before the table definitions, there is a double with the number of tables defined. Different table definitions are bracketed by curly braces ('{' and '}'), and within each table definition there are multiple delimiters.

The definition of each user-defined table has the following keywords:

Order	Keyword	Description	Value - C Language Type
1	tid	User-defined Table_ID	Long integer
2	description	Description of the data in the table	Character Array
3	number_ind	Number of independent arrays (nr, nt, etc)	Long integer
4	num_arrays	Total number of arrays (including number_ind)	Long integer
5	nr	Value of nr	Long integer
6	nt	Value of nt	Long integer
7	sizes	Sizes of each array	Character Array(s)
8	labels	Labels for each array	Character Array(s)

Most keywords are followed by a space, a colon, a space, the field's value, a space, and then comma. Character arrays are delimited with double quotes. The keywords **sizes** and **labels** may contain multiple values, and they have a complex delimiting structure described below. Here is an outline of each keyword and its structure:

1. **Tid** is the only keyword that is not followed by a space; a colon is immediately after the keyword. Its value is a C language long integer and must be a non-SES_IO defined Table_ID. Note: if the value is larger than 999999, the data cannot be converted into SESAME ASCII 1 file format. SESAME ASCII 1's Table_ID is restricted to 6 characters.
   An example of a tid:
   tid: 123456 ,

2. **Description** is used to describe the data in this user-defined table. The value is an array of characters that is delimited by double quotes. This value must not contain the @ character.
   An example:
   description : "A description of the data in this table" ,

3. **Number_ind** is the number of independent arrays. An independent array is a misnomer, since each of these arrays is used to hold scalar value. These values are in turn used to describe the sizes of each array. This value contains a long integer.
   An example:
   number_ind : 2 ,
   Generally, there are at least 2 independent arrays, one each for nr and nt. There must always be a nr and nt.

4. **Num_arrays** are the number of arrays in the table's data. This value includes the scaler arrays defined in number_ind. As an illustration, if number_ind = 2 and num_arrays = 3, two of the arrays are independent. This value contains a long integer.
   An example
   num_arrays : 6 ,

5. **Nr** is the number of densities, however nr is actually just a variable name and may stand for any item you wish. Nr is used to calculate the sizes of data arrays and contains a long integer. An example:
   nr : 12 ,

6. **Nt** is the number of temperatures, however nt is actually just a variable name and may stand for any item you wish. Nt is used to calculate the sizes of data arrays and contains a long integer. An example:
   nt : 8 ,

7. **Sizes** describes the sizes of all of the arrays in the data. The number of entries in sizes must be the same as num_arrays. The independent array descriptions are first, followed by the remaining arrays. The first two descriptions are always for nr and nt. These values are always scalar and must contain the value 1. The rest of the arrays can be defined by using the value of one of the independent arrays. The value of each entry is delimited by parenthesis, while the entire section of data is bracketed by square brackets. An example:
   sizes : [(1) , (1) , (nr) , (nt) , (nr*nt) , (2*nt)] ,

8. **Labels** are character arrays which describe each array. There must be the same number of character arrays as there are num_arrays. None of these values may contain the @ character. The value of each entry is delimited by double quotes, while the entire section of data is bracketed by square brackets. An example:
   labels : [“nr (number of densities)” , “nt (number of temperatures)” , “description for array 3” , “description for array 4” , “description for array 5” , “description for array 6”]

Please note: User-defined tables are for a quick prototype and nothing more. These structures are used internally is SES_IO and may look non-sensical. User-defined tables are limited to default SES_IO table and grid definitions. This means that user-defined tables must contain at least 2 number_ind arrays for nr & nt, where nr is defined first and nt second in the data. The values of nr and nt must be greater than 0, else SES_IO’s ses_setup fails. Even if you do not plan on using one or the other value, it must be defined. Remember this is for quick, and possibly non-accurate prototyping.

If you want to add a new table to SES_IO’s code, the files to focus on are: internals/_ses_standard.c, internals/_ses_table.c, internals/_ses_table_iterator.c, internals/*/_get_grid_*.c (where * = binary, ascii, ascii2, xml, etc).

The following is an example of what a 100 table looks like:


```
1 12340 100 912 20210217 20210217 1
  2.000000000000000E+00 { tid: 632  , description : "Shear Modulus Table with Melt Correlation" , numbe
r_ind : 3 , num_arrays : 8 , nr : 4 , nt : 1 , sizes : [  (1) , (1) , (1) , (nr)
 , (nt) , (nr) , (nr) , (np)  ] , labels : [ "nr (number densities)" , "nt = 1(n
umber temperatures)" , "np (number pressure)" , "r - density (Mg/m3)" , "t = 0 t
emperature (K)" , "g - shear modulus (GPa)" , "g - shear modulus (GPa) at melt"
, "g - shear modulus (GPa) at vapor" ] } @@@@@@@ { tid: 756  , description : "Sh
ear Modulus Table" , number_ind : 2 , num_arrays : 5 , nr : 6 , nt : 3 , sizes :
 [  (1) , (1) , (rho) , (kelvin) , (rho)  ] , labels : [ "rho (number densities)
" , "kelvin (number temperatures)" , "r - density (Mg/m3)" , "t = 0 temperature
(K)" , "g - shear modulus (GPa)" ] } @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@
```

6.0 Examples of SESAME ASCII 2 File

This is an example of a Sesame ASCII Version 2 file with two tables:

```
version 2.0
0 3720 101 139 19931221 19940607 1
material. Al\ source. Katz Malone\ date. Dec. 21 1993\ refs. LA-UR-92-1307\ comp
. Al\ codes. OpenSesame 2.2\ classification. unclassified \
1 3720 201 5 19931221 19940607 1
1.99999989E+01 4.00799980E+01 1.54699976E+00 0.00000000E+00 0.00000000E+00
```

Another example with two user-defined tables:

```
Version 2.0
 0 12340 632 21 20130918 20190423 3
  4.000000000000000E+00  1.000000000000000E+00  5.000000000000000E+00  0.000000000000000E+00  2.000000000000000E+00
  4.000000000000000E+00  6.000000000000000E+00  0.000000000000000E+00  0.000000000000000E+00  1.000000000000000E+00
  2.000000000000000E+00  3.000000000000000E+00  4.000000000000000E+00  3.000000000000000E+00  2.000000000000000E+00
  1.000000000000000E+00  6.000000000000000E+00  5.000000000000000E+00  4.000000000000000E+00  3.000000000000000E+00
  2.000000000000000E+00
 1 12340 756 17 20130918 20190423 3
  6.000000000000000E+00  3.000000000000000E+00  0.000000000000000E+00  1.000000000000000E+00  2.000000000000000E+00
  3.000000000000000E+00  4.000000000000000E+00  5.000000000000000E+00  0.000000000000000E+00  0.000000000000000E+00
  0.000000000000000E+00  0.000000000000000E+00  4.000000000000000E+00  8.000000000000000E+00  1.200000000000000E+01
  1.600000000000000E+01  2.000000000000000E+01
 1 12340 100 912 20210217 20210217 1
  2.000000000000000E+00 { tid: 632  , description : "Shear Modulus Table with Melt Correlation" , numbe
r_ind : 3 , num_arrays : 8 , nr : 4 , nt : 1 , sizes : [  (1) , (1) , (1) , (nr)
 , (nt) , (nr) , (nr) , (np)  ] , labels : [ "nr (number densities)" , "nt = 1(n
umber temperatures)" , "np (number pressure)" , "r - density (Mg/m3)" , "t = 0 t
emperature (K)" , "g - shear modulus (GPa)" , "g - shear modulus (GPa) at melt"
, "g - shear modulus (GPa) at vapor" ] } @@@@@@@ { tid: 756  , description : "Sh
ear Modulus Table" , number_ind : 2 , num_arrays : 5 , nr : 6 , nt : 3 , sizes :
 [  (1) , (1) , (rho) , (kelvin) , (rho)  ] , labels : [ "rho (number densities)
" , "kelvin (number temperatures)" , "r - density (Mg/m3)" , "t = 0 temperature
(K)" , "g - shear modulus (GPa)" ] } @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@
```